*Journal of the Nigerian Association of Mathematical Physics*
*Volume* **14 (May, 2009), pp 349 – 358**


# Comparison of second and third orders Runge-Kutta methods for solving initial-value problems in ordinary differential equations

**[1]L. A. Ukpebor, [2]A. C. Esekhaigbe and [3]E. Okodugha**
[1]*Depaartment of Mathematics, Ambrose Alli University, Ekpoma.*
[2]*Department of Statistics, Auchi Polytechnic, Auchi.*
[3]*Department of Pre-ND Sciences, Auchi Polytechnic, Auchi.*

***Abstract***

***This work is concerned with the analysis of second and third orders Runge-Kutta formulae capable of solving initial value problems in Ordinary Differential Equations of the form: $y^1 = f(x, y)$, $y(x_0) = y_0$, $a \le x \le b$. The intention is to find out which of these two orders can improve the performance of results when implemented on the initial-value problems defined above. We found out that the higher the order, the better the performance of that order. When parameters are properly varied, performance may also improve.***

## 1.0 Introduction

The object of this research work is to use Runge-Kutta Methods to solve initial value problems in Ordinary Differential Equations of the form: $y^1 = f(x, y)$, $y(x_0) = y_0$, $a \le x \le b$, with a view to finding out its consistency and convergence.

In the work of Lambert [1], the Runge-Kutta Methods represent an important family of implicit and explicit iterative methods for approximation of ordinary differential equations in numerical analysis. Furthermore, of all computational methods for the numerical solution of initial-value problems, the easiest to implement is Euler's rule. It is explicit and, being a one-step method, requires no additional starting values and readily permits a change of step length during the computation. Its low order, of course, makes it of limited practical value. Linear multi-step methods achieve higher order by sacrificing the one-step nature of the algorithm, while retaining linearity with respect to $y_{n+j}, f_{n+j}, j = 0,1,\ldots,k$. Higher order can also be achieved by sacrificing linearity, but preserving the one-step nature of the algorithm.

Thus, the philosophy behind the Runge-Kutta method is to retain the advantages of one-step methods. Due to the loss of linearity, error analysis is considerably more difficult than in the case of linear multi-step methods. Traditionally, Runge-Kutta methods are all explicit, although, recently, implicit Runge-Kutta Method, which have improved weak stability characteristics, have been considered.

We shall use the phrase "Runge-Kutta Method" to mean explicit Runge-Kutta Method. Thus, a Runge-Kutta method may be regarded as a particular case of the general explicit one-step method of the kind:

$$y_{n+1} - y_n = h\,\Phi(x_n, y_n; h). \qquad (1.1)$$

### 1.1 The statement of the problem

We are interested in solving the general initial-value problem:

$$y^1 = f(x, y),\ y(x_0) = y_0,\ a \le x \le b,\ y(0) \text{ (given)} \qquad (1.2)$$

**[1]Corresponding author:**
**[1]Telephone: 08033688271**

(Whose gradient function $f(x, y)$ may have points of discontinuities in a differential system called points of singularities).

## 2.0 The second-order Runge-Kutta methods

The general R-stage Runge-Kutta method is:

$$y_{n+1} - y_n = h\phi(x_n, y_n; h),$$

$$\phi(x_n, y_n; h) = \sum_{r=1}^{R} c_r k_r,$$

$$k_1 = f(x,y), \tag{2.3}$$

$$k_r = f(x + ha_r, y + h\sum_{s=1}^{r-1} b_{rs} k_s), \; r = 2, 3, ..., R,$$

$$a_r = \sum_{s=1}^{r-1} b_{rs}, \; r = 2, 3, \ldots, R$$

Now, for the case $R = 2$, we shall have:

$$k_1 = hf(x_n, y_n)$$

$$k_2 = hf(x_n + a_2 h, y_n + hb_{21}k_1) \tag{2.4}$$

and $y_{n+1} = y_n + c_1k_1 + c_2k_2$, where the parameters $a_2$, $b_{21}$, $c_1$ and $c_2$ are chosen to make $y_{n+1}$ close to $y(x_{n+1})$, $k_1 = hf_n$. Expanding $k_2$ as a Taylor's series about the point (x, y) we have:

$k_2 = hf(x_n + a_2h, y_n + b_{21}hf_n) = h[f(x_n, y_n) + (a_2hf_x + b_{21}hf_nf_y) + {}^1/_{2!}(a_2^2h^2f_{xx} + 2a_2b_{21}h^2f_nf_{xy} + b_{21}^2h^2f_n^2f_{yy}) + \ldots]$

$k_{2n} = hf_n + h^2(a_2f_x + b_{21}f_nf_y) + h^3/_2(a_2^2f_{xx} + 2a_2b_{21}f_nf_{xy} + b_{21}^2f_n^2f_{yy}) + \ldots$

Substituting the value of $k_1$ and $k_2$ into equation (2.2), we get,

$$y_{n+1} = y_n + (c_1 + c_2)hf_n + h^2(a_2c_2f_x + b_{21}c_2f_nf_y) + h^3/_2\, c_2(a_2^2f_{xx} + 2a_2b_{21}f_nf_{xy} + b_{21}^2f_n^2f_{yy}) + \ldots \tag{2.5}$$

The Taylor's series about the point $(x, y)$ is:

$$y(x_{n+1}) = y(x_n) + hy^1(x_n) + h^2/_{2!}\, y^{11}(x_n) + h^3/_{3!}y^{\prime\prime\prime}(x_n) + \ldots \tag{2.6}$$

where,

$y' = f(x, y)$

$y'' = f' = f_x + f_y y' = f_x + f_y f_x$

$y''' = f'' = f_{xx} + f_{xy}f + f_{yx}f + f_{yy}f^2 + f_yf_x + f_y^2 = f_{xx} + 2f_{xy}f + f_{yy}f^2 + f_xf_y + f_y^2f$

$y^{(iv)} = f''' = f_{xxx} + 3ff_{xxy} + 3f^2f_{xyy} + 3f_xf_{xy} + 5ff_{xy}f_y + 3ff_xf_{yy} + f_{xx}f_y + 4f^2f_yf_{yy} + f_xf_y^2 + ff_y^3 + f^3f_{yyy}$

$y^{(v)} = f^{iv} = f_{xxxx} + 4ff_{xxxy} + 6f^2f_{xxyy} + 6f_xf_{xxy} + 9ff_{xxy}f_y + 4f^3f_xf_{yyy} + 6f_{xx}f_{xyy} + 15f^2f_{xyy}f_y + 6ff_xf_{xyy} + 4f_{xx}f_{xy} + 8ff^2f_{xy} + 8f^2f_{xy}f_{yy} + 7f_xf_y + 9ff_{xy}f_y^2 + 6f^2f_xf_{yyy} + 4ff_{xx}f_{yy} + 3f_x^2f_{yy} + 13ff_xf_yf_{yy} + f_{xxx}f_y + 7f^3f_yf_{yyy} + 4f^3f_{yy}^2 + 4f^2f_{xy}f_{yy} + 11f^2f_y^2f_{yy} + f_{xx}f_y^2 + f_xf_y^2 + f_xf_y^3 + ff_y^4 + f^4f_{yyyy},$

where $f_x$, $f_y$ represent the derivatives of $f$ with respect to $x$,$y$ respectively.

Comparing equations (2.5) and (2.6) and matching coefficients of powers of h, we obtain three equations for the parameters,

$$c_1 + c_2 = 1$$

$$a_2c_2 = ½ \tag{2.7}$$

$$b_{21}c_2 = ½$$

From these equations, we see that if $c_2$ is chosen arbitrarily (non zero), then

$$b_{21} = a_2, \; c_2 = {}^1/2a_2, \; c_1 = 1 - {}^1/2a_2) \tag{2.8}$$

using (2.8) and (2.5), we get

$$y_{n+1} = y_n + hf_n + h^2/_2(f_x + f_nf_y) + c_2h^3/_4(f_{xx} + 2f_nf_{xy} + f_n^2f_{yy}) + \ldots \tag{2.9}$$

Subtracting (2.9) from (2.5), we obtain the local truncation error

$$T_n = y(x_{n+1}) - y_{n+1}$$

$$T_n = h^3[({}^1/_6 - a_2/_4)(f_{xx} + 2f_nf_{xy} + f_n^2f_{yy}) + {}^1/_6f_y(f_x + f_nf_y)] + \ldots$$

We observe that no choice of the parameter $a_2$ will make the leading term $T_n$ vanish for all $f(x,y)$. The local truncation error depends not only on derivatives of the solution $y(x)$ but also

on the function *f*. This is typical of all Runge-Kutta methods, in most other methods the truncation error depends only on certain derivatives of $y(x)$. Generally, $a_2$ will be chosen between 0 and 1. Sometimes, the free parameters are chosen either to have as large as possible the interval of absolute stability or to minimize the sum of the absolute values of the coefficients in the term $T_n$. Such a choice is called optimal.

Thus, we state the second order Runge-Kutta methods by listing the coefficients as follows:

| $a_2, b_{21}$ | $c_1, c_2$ |
|---|---|

(a)

| ½, ½ | |
|---|---|
| | 0, 1 |

(b).

| $^2/_3, ^2/_3$ | |
|---|---|
| | ¼, ¾ |

(c).

| 1, 1 | |
|---|---|
| | ½, ½ |

These give the following formulae respectively:

(a) $k_1 = hf(x_n, y_n)$
$k_2 = hf(x_n, ½h, y_n + ½k_1)$
$y_{n+1} = y_n + k_2$

(b) $k_1 = hf(x_n, y_n)$
$k_2 = hf(x_n, {}^2/_3 h, y_n + {}^2/_3 k_1)$
$y_{n+1} = y_n + ¼k_1 + ¾k_2$

(c) $k_1 = hf(x_n, y_n)$
$k_2 = hf(x_n, h, y_n + k_1)$
$y_{n+1} = y_n + ½k_1 + ½k_2$

## 2.1 The third-order Runge-Kutta methods

We shall consider the derivation of a third order Runge-Kutta method. With reference to Lambert [1] and Jain [2], the third-order Runge-Kutta Method is derived thus:

At the stage, $R = 3$.The method is derived from the general Runge-Kutta formula defined above. Hence,

$$\phi(x_n, y_n, h) = c_1k_1 + c_2k_2 + c_3k_3$$
$$k_1 = f(x_n, y_n), \qquad (2.10)$$
$$k_2 = f(x_n + ha_2, y_n + h(b_{21}k_1)$$
$$k_3 = f(x_n + ha_3, y_n + h(b_{31}k_1 + b_{32}k_2))$$

and $a_2 = b_{21}$, $a_3 = b_{31} + b_{32} \Rightarrow b_{31} = a_3 - b_{32}$

Thus, $y_{n+1} - y_n = h(c_1k_1 + c_2k_2 + c_3k_3)$

with
$$k_1 = f(x_n, y_n)$$
$$k_2 = f(x_n + ha_2, y_n + ha_2k_1) \qquad (2.11)$$

and $k_3 = f(x_n + ha_3, y_n + h(a_3 - b_{32})k_1 + hb_{32}k_2)$

Expanding $k_2$ as a Taylor's series about the point $(x, y)$ we have:

$$k_2 = \sum_{r=0}^{\infty} \frac{1}{r!} \left[ \left( ha_2 \frac{\partial}{\partial x} + ha_2k_1 \frac{\partial}{\partial x} \right)^r f(x, y) \right]$$

Hence, $k_2 = f + ha_2(f_x + k_1f_y) + h^2/_2 a_2^2(f_{xx} + 2k_1f_{xy} + k_1^2f_{yy}) + 0(h^3)$. Similarly,

$$k_3 = \sum_{r=0}^{\infty} \frac{1}{r!}\left[\left( ha_3 \frac{\partial}{\partial x} + h[(a_3 - b_{32})k_1 + hb_{32}k_2]\frac{\partial}{\partial x}\right)^r f(x,y)\right]$$

$$k_3 = f(x_n, y_n) + ha_3 \frac{\partial}{\partial x} f(x_n, y_n) + h[(a_3 - b_{32})k_1 + hb_{32}k_2] \frac{\partial}{\partial x} f(x_n, y_n) + \tfrac{1}{2}[ha_3 \frac{\partial}{\partial x} + h[(a_3 - b_{32})k_1 + hb_{32}k_2] \frac{\partial}{\partial x}]^2 \frac{\partial^2}{\partial x dy} f(x_n, y_n) + 0(h^3)$$

$$k_3 = f(x_n, y_n) + h[a_3 f_x + [(a_3 - b_{32})k_1 b_{32}k_2]fy] + \frac{h^2}{2}[a_3^2 f_{xx} + a_3[(a_3 - b_{32})k_1 + b_{32}k_2]f_{xy} + [(a_3 - b_{32})k_1 + b_{32}k_2]^2 f_{yy}] + 0(h^3)$$

substituting $k_1$, $k_2$ and $k_3$ into equation (2.8), we have $\phi(x_n, y_n, h) = c_1k_1 + c_2k_2 + c_3k_3$ becoming,

$$y_{n+1} - y_n = h(c_1+c_2+c_3)f + h^2(c_2a_2 + c_3a_3)f_x + h^2(c_2a_2k_1 + c_3a_3k_1)f_y + \tfrac{1}{2}h^3[c_2a_3^2 + c_3a_3^2k_1^2 - 2c_3a_3b_{32}k_1^2 + 2c_2a_2b_{32}k_1^2 - c_3b_{32}^2k_1^2 + c_2b_{32}^2k_1^2] f_{yy} + 0(h^4) \quad (2.12)$$

Now, comparing this with the Taylor's series expansion for order 3, we have,

$$y(x_n) - y(x_0) = hk_1 + h^2/_2 f_x + h^2/_2 k_1 f_y + h^3/_6 f_{xx} + h^3/_6 f_x f_y + h^3/_3 k_1 f_{xy} + h^3/_6 k_1 f_y^2 + h^3/_6 k_1^2 f_{yy} \quad (2.13)$$

by equating coefficients, we have

$$\begin{aligned} c_1 + c_2 + c_3 &= 1 \\ c_2a_2 + c_3a_3 &= \tfrac{1}{2} \\ c_2a_2^2 + c_3a_3^2 &= {}^1/_3 \\ c_3a_2b_{32} &= {}^1/_6 \end{aligned} \quad (2.14)$$

To this, we have four equations in six unknowns and there exists two parameter family of solutions.

This two particular solutions lead to well-known third order Runge-Kutta Methods.

(a) $c_1 = \tfrac{1}{4}$, $c_2 = 0$, $c_3 = \tfrac{3}{4}$, $a_2 = {}^1/_3$, $a_3 = {}^2/_3$, $b_{32} = {}^2/_3$. The resulting method may be written:

$$\begin{aligned} y_{n+1} - y_n &= {}^h/_4 (k_1 + 3k_3), \\ k_1 &= f(x_n, y_n), \\ k_2 &= f(x_n + {}^1/_3 h, y_n + {}^1/_3 hk_1) \\ k_3 &= f(x_n + {}^2/_3 h, y_n + {}^2/_3 hk_2) \end{aligned} \quad (2.15)$$

Equation (2.15) is known as the Heun's third-order formula. We observe that the computational advantages in choosing $c_2 = 0$ is somewhat illusory since, although $k_2$ does not appear in the first equation of equation(s). It must nevertheless be calculated at each step.

(b) $c_1 = {}^1/_6$, $c_2 = {}^2/_3$, $c_3 = {}^1/_6$, $a_2 = \tfrac{1}{2}$, $a_3 = 1$, $b_{32} = 2$. The resulting method,

$$\begin{aligned} y_{n+1} - y_n &= {}^h/_6 (k_1 + 4k_3 + k_3), \\ k_1 &= f(x_n, y_n), \\ k_2 &= f(x_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}hk_1) \\ k_3 &= f(x_n + h, y_n - hk_1 + 2hk_2) \end{aligned} \quad (2.16)$$

is Kutta's third-order rule. It is the most popular third-order Runge-Kutta Method for desk computation (largely because the coefficient $\tfrac{1}{2}$ is preferable to ${}^1/_3$, which appears frequently in equation (2.15).

## 3.0 Numerical computations and results

In order to access the performance of the second and third orders Runge-Kutta methods, the following sample problems were solved:

(i) $y^1 = 1 + y^2$, $y(0) = 1$, $0 \le x \le 1$, whose theoretical solution is: $y(x) = \tan(x + {}^\Pi/_4)$

(ii) $y^1 = -y$, $y(0) = 1$, $0 \le x \le 1$, whose theoretical solution is: $y(x) = {}^1/_{\exp x}$

(iii) $y^1 = y$, $y(0) = 1$, $0 \le x \le 1$, whose theoretical solution is: $y(x) = expx$

(iv) $y^1 = y^2$, $y(0) = 1$, $0 \le x \le 1$, whose theoretical solution is: $y(x) = {}^1/_{1-x}$

## 3.1 Tables of results

**Order 2 Problem 1**

| XN | YN | TSOL | ERROR |
|---|---|---|---|
| . 1D+00 | 0.1222000000000D+01 | 0.1223048880450D+01 | 0.1048880449865D-02 |
| .2D+00 | 0.1504904563032D+01 | 0.1508497647121D+01 | 0.3593084089073D-02 |
| .3D+00 | 0.1885838780962D+01 | 0.1895765122854D+01 | 0.9926341892099D-02 |
| .4D+00 | 0.2437784037299D+01 | 0.2464962756723D+01 | 0.2717871942365D-01 |
| .5D+00 | 0.3325414563118D+01 | 0.3408223442336D+01 | 0.8280887921824D-01 |
| .6D+00 | 0.5004946244862D+01 | 0.5331855223459D+01 | 0.3269089785964D+00 |
| .7D+00 | 0.9252945637605D+01 | 0.1168137380031D+02 | 0.2428428162705D+01 |
| .8D+00 | 0.2968052273478D+02 | -.6847966834558D+02 | -.9816019108036D+02 |
| .9D+00 | 0.7685396051645D+03 | -.8687629546482D+01 | -.7772272347110D+03 |
| .1D+01 | 0.1790353923463D+09 | -.4588037824984D+01 | -.1790353969344D+09 |

**Order 3 Problem 1**

| XN | YN | TSOL | ERROR |
|---|---|---|---|
| .1D+00 | 0.1222901142387D+01 | 0.1223048880450D+01 | 0.1477380630341D-03 |
| .2D+00 | 0.1507980352553D+01 | 0.1508497647121D+01 | 0.5172945686114D-03 |
| .3D+00 | 0.1894251391813D+01 | 0.1895765122854D+01 | 0.1513731041409D-02 |
| .4D+00 | 0.2460382773866D+01 | 0.2464962756723D+01 | 0.4579982856682D-02 |
| .5D+00 | 0.3391873333485D+01 | 0.3408223442336D+01 | 0.1635010885070D-01 |
| .6D+00 | 0.5248451742415D+01 | 0.5331855223459D+01 | 0.8340348104371D-01 |
| .7D+00 | 0.1069156363364D+02 | 0.1168137380031D+02 | 0.9898101666695D+00 |
| .8D+00 | 0.5993762963300D+02 | -.6847966834558D+02 | -.1284172979786D+03 |
| .9D+00 | 0.3675614567764D+06 | -.8687629546482D+01 | -.3675701444059D+06 |
| .1D+01 | 0.1371434973316D+36 | -.4588037824984D+01 | -.1371434973316D+36 |

**Order 2 Problem 2**

| XN | YN | TSOL | ERROR |
|---|---|---|---|
| .1D+00 | 0.9050000000000D+00 | 0.9048374180360D+00 | -.1625819640404D-03 |
| .2D+00 | 0.8190250000000D+00 | 0.8187307530780D+00 | -.2942469220182D-03 |
| .3D+00 | 0.7412176250000D+00 | 0.7408182206817D+00 | -.3994043182819D-03 |
| .4D+00 | 0.6708019506250D+00 | 0.6703200460356D+00 | -.4819045893606D-03 |
| .5D+00 | 0.6070757653156D+00 | 0.6065306597126D+00 | -.5451056029917D-03 |
| .6D+00 | 0.5494035676106D+00 | 0.5488116360940D+00 | -.5919315166144D-03 |
| .7D+00 | 0.4972102286876D+00 | 0.4965853037914D+00 | -.6249248962205D-03 |
| .8D+00 | 0.4499752569623D+00 | 0.4493289641172D+00 | -.6462928450836D-03 |
| .9D+00 | 0.4072276075509D+00 | 0.4065696597406D+00 | -.6579478102870D-03 |
| .1D+01 | 0.3685409848336D+00 | 0.3678794411714D+00 | -.6615436621096D-03 |

**Order 3 Problem 2**

| XN | YN | TSOL | ERROR |
|---|---|---|---|
| .1D+00 | 0.9048333333333D+00 | 0.9048374180360D+00 | 0.4084702626250D-05 |
| .2D+00 | 0.8187233611111D+00 | 0.8187307530780D+00 | 0.7391966870607D-05 |
| .3D+00 | 0.7408081879120D+00 | 0.7408182206817D+00 | 0.1003276968092D-04 |
| .4D+00 | 0.6703079420291D+00 | 0.6703200460356D+00 | 0.1210400656437D-04 |
| .5D+00 | 0.6065169695460D+00 | 0.6065306597126D+00 | 0.1369016665864D-04 |
| .6D+00 | 0.5487967712775D+00 | 0.5488116360940D+00 | 0.1486481651014D-04 |
| .7D+00 | 0.4965696118776D+00 | 0.4965853037914D+00 | 0.1569191380352D-04 |
| .8D+00 | 0.4493127371473D+00 | 0.4493289641172D+00 | 0.1622696996778D-04 |
| .9D+00 | 0.4065531416621D+00 | 0.4065696597406D+00 | 0.1651807852571D-04 |
| .1D+01 | 0.3678628343472D+00 | 0.3678794411714D+00 | 0.1660682420951D-04 |

**Order 2 Problem 3**

| XN | YN | TSOL | ERROR |
|---|---|---|---|
| .1D+00 | 0.1105000000000D+01 | 0.1105170918076D+01 | 0.1709180756475D-03 |
| .2D+00 | 0.1221025000000D+01 | 0.1221402758160D+01 | 0.3777581601698D-03 |
| .3D+00 | 0.1349232625000D+01 | 0.1349858807576D+01 | 0.6261825760028D-03 |
| .4D+00 | 0.1490902050625D+01 | 0.1491824697641D+01 | 0.9226470162702D-03 |
| .5D+00 | 0.1647446765941D+01 | 0.1648721270700D+01 | 0.1274504759503D-02 |
| .6D+00 | 0.1820428676364D+01 | 0.1822118800391D+01 | 0.1690124026118D-02 |
| .7D+00 | 0.2011573687383D+01 | 0.2013752707470D+01 | 0.2179020087825D-02 |
| .8D+00 | 0.2222788924558D+01 | 0.2225540928492D+01 | 0.2752003934638D-02 |
| .9D+00 | 0.2456181761636D+01 | 0.2459603111157D+01 | 0.3421349520548D-02 |
| .1D+01 | 0.2714080846608D+01 | 0.2718281828459D+01 | 0.4200981850821D-02 |

**Order 3 Problem 3**

| XN | YN | TSOL | ERROR |
|---|---|---|---|
| .1D+00 | 0.1105166666667D+01 | 0.1105170918076D+01 | 0.4251408980860D-05 |
| .2D+00 | 0.1221393361111D+01 | 0.1221402758160D+01 | 0.9397049058668D-05 |
| .3D+00 | 0.1349843229588D+01 | 0.1349858807576D+01 | 0.1557798803975D-04 |
| .4D+00 | 0.1491801742566D+01 | 0.1491824697641D+01 | 0.2295507497307D-04 |
| .5D+00 | 0.1648689559160D+01 | 0.1648721270700D+01 | 0.3171154060877D-04 |
| .6D+00 | 0.1822076744464D+01 | 0.1822118800391D+01 | 0.4205592604678D-04 |
| .7D+00 | 0.2013698482091D+01 | 0.2013752707470D+01 | 0.5422537983479D-04 |
| .8D+00 | 0.2225472439124D+01 | 0.2225540928492D+01 | 0.6848936862669D-04 |
| .9D+00 | 0.2459517957305D+01 | 0.2459603111157D+01 | 0.8515385191776D-04 |
| .1D+01 | 0.2718177262482D+01 | 0.2718281828459D+01 | 0.1045659774341D-03 |

**Order 2 Problem 4**

| XN | YN | TSOL | ERROR |
|---|---|---|---|
| .1D+00 | 0.1110500000000D+01 | 0.1111111111111D+01 | 0.6111111111111D-03 |
| .2D+00 | 0.1248276228587D+01 | 0.1250000000000D+01 | 0.1723771413397D-02 |
| .3D+00 | 0.1424760126021D+01 | 0.1428571428571D+01 | 0.3811302550067D-02 |
| .4D+00 | 0.1658736394656D+01 | 0.1666666666667D+01 | 0.7930272010906D-02 |
| .5D+00 | 0.1983300735783D+01 | 0.2000000000000D+01 | 0.1669926421672D-01 |
| .6D+00 | 0.2462397829859D+01 | 0.2500000000000D+01 | 0.3760217014070D-01 |
| .7D+00 | 0.3236425671191D+01 | 0.3333333333333D+01 | 0.9690766214193D-01 |
| .8D+00 | 0.4677725672023D+01 | 0.5000000000000D+01 | 0.3222743279767D+00 |
| .9D+00 | 0.8128767717881D+01 | 0.1000000000000D+02 | 0.1871232282119D+01 |

**Order 3 Problem 4**

| XN | YN | TSOL | ERROR |
|---|---|---|---|
| .1D+00 | 0.1111057827572D+01 | 0.1111111111111D+01 | 0.5328353909473D-04 |
| .2D+00 | 0.1249840436239D+01 | 0.1250000000000D+01 | 0.1595637609275D-03 |
| .3D+00 | 0.1428192621392D+01 | 0.1428571428571D+01 | 0.3788071794708D-03 |
| .4D+00 | 0.1665807462331D+01 | 0.1666666666667D+01 | 0.8592043360645D-03 |
| .5D+00 | 0.1997985670073D+01 | 0.2000000000000D+01 | 0.2014329926953D-02 |
| .6D+00 | 0.2494791095604D+01 | 0.2500000000000D+01 | 0.5208904395528D-02 |
| .7D+00 | 0.3317137851229D+01 | 0.3333333333333D+01 | 0.1619548210399D-01 |
| .8D+00 | 0.4929048284012D+01 | 0.5000000000000D+01 | 0.7095171598811D-01 |
| .9D+00 | 0.9343550594338D+01 | 0.1000000000000D+02 | 0.6564494056619D+00 |

The computer program for the results in this paper can be found in the appendix.

## 4.0 Findings

After the implementation of the second and third orders Runge-Kutta formulae on some selected initial-value problems, it was discovered from our tables of results that the third-

order Runge-Kutta formula has higher accuracy from the error column, the accuracy and effectiveness of the third-order formula is also seen.

One can say that the higher the order (R), the higher the accuracy of that order. But when parameters are changed to high grade, the formula may produce more accurate results. We also found out that when we compared our expansion with the Taylor series expansion, we got some non-linear equations.

The parameters in these non-linear equations exceed the number of equations but in the work of Lambert [1], these parameters are chosen to ensure that the method has:

(i) Adequate order of accuracy
(ii) Minimum bound of local truncation error
(iii) Minimum computer storage facilities.

## 5.0 Conclusion

Finally, it is clear that the survey carried out is of high value to see the effectiveness and accuracy of the second-order and third-order Runge-Kutta formulae in handling initial-value problems in Ordinary Differential Equations. This analysis proved that when the method is of higher order, it may likely produce more effective and accurate results if parameters are properly varied.

### 5.1 Summary

Summarily, our $k_{i,s}$ where expanded by Taylor series expansion about the point $(x_n\ y_n)$. These expansions are inserted into $\phi(x_n, y_n; h)$ and compared with the Taylor series expansion of $\phi(x_n, y_n; h)$.This comparism resulted in some non-linear equations whose parameters were chosen to ensure that the equations were satisfied.

Hence, the Runge-Kutta method as a single step method has shown that no area of research is over exhausted depending on the area of interest.

**APPENDIX**
**FORTRAN PROGRAMS THAT GENERATED THE RESULTS.**

```
C PRO
C SECOND-ORDER  RUNGE-KUTTA METHOD
C OUR PROBLEM IS :Y'=1+Y**2,Y(O)=1
C THEORETICAL SOLUTION:Y(XN)=TAN(XN+PI/4)
      DOUBLE PRECISION XN,YN,H,PI,ONE,TWO,FOUR
      DOUBLE PRECISION TSOL,ERROR,K1,K2
      OPEN(6,FILE='RUNG2.OUT')
      H=0.1D0
      YN=1.0D0
      XN=0.1D0
      ONE=1.0D0
      TWO=2.0D0
      FOUR=4.0D0
       PI=FOUR*DATAN(ONE)
      WRITE(6,101)
3      K1=ONE+YN**2
      K2=ONE+(YN+H*K1)**2
      YN=YN+H/TWO*(K1+K2)
       TSOL=TAN(XN+(PI/FOUR))
       ERROR=TSOL-YN
       WRITE(6,100)XN,YN,TSOL,ERROR
      XN=XN+H
      IF(XN.LE.ONE) GOTO 3
 100  FORMAT(D6.1,1X,3(3X,D19.13))
 101  FORMAT(2X,'XN',13X,'YN',16X,'TSOL',16X,'ERROR')
      END
C PRO
C THIRD- ORDER  RUNGE-KUTTA METHOD
```

```
C OUR PROBLEM IS :Y'=1+Y**2,Y(O)=1
C THEORETICAL SOLUTION:Y(XN)=TAN(XN+PI/4)
      DOUBLE PRECISION XN,YN,H,PI,ONE,THREE,TWO,FOUR
      DOUBLE PRECISION TSOL,ERROR,K1,K2,K3
      OPEN(6,FILE='RUNG2.OUT')
       H=0.1D0
      YN=1.0D0
      XN=0.1D0
      HALF=0.5D0
       THREE=3.0D0
      ONE=1.0D0
      TWO=2.0D0
      FOUR=4.0D0
      PI=FOUR*DATAN(ONE)
      WRITE(6,101)
3     K1=ONE+YN**2
      K2=ONE+(YN+(H/THREE)*K1)**2
      K3=ONE+(YN+(H/THREE)*TWO*K2)**2
      YN=YN+H/FOUR*(K1+THREE*K3)
      TSOL=TAN(XN+(PI/FOUR))
      ERROR=TSOL-YN
      WRITE(6,100)XN,YN,TSOL,ERROR
      XN=XN+H
      IF(XN.LE.ONE) GOTO 3
100   FORMAT(D6.1,1X,3(3X,D19.13))
101   FORMAT(2X,'XN',13X,'YN',16X,'TSOL',16X,'ERROR')
      END

C PRO
C SECOND-ORDER  RUNGE-KUTTA METHOD
C OUR PROBLEM IS :Y'=-Y,Y(O)=1
C THEORETICAL SOLUTION:Y(XN)=1/EXP(XN)
      DOUBLE PRECISION XN,YN,H,ONE,TWO
      DOUBLE PRECISION TSOL,ERROR,K1,K2
      OPEN(6,FILE='RUNG2.OUT')
      H=0.1D0
      YN=1.0D0
      XN=0.1D0
      ONE=1.0D0
      TWO=2.0D0
      WRITE(6,101)
 3    K1=-YN
      K2=-(YN+H*K1)
      YN=YN+H/TWO*(K1+K2)
      TSOL=ONE/EXP(XN)
      ERROR=TSOL-YN
      WRITE(6,100)XN,YN,TSOL,ERROR
      XN=XN+H
      IF(XN.LE.ONE) GOTO 3
100   FORMAT(D6.1,1X,3(3X,D19.13))
101   FORMAT(2X,'XN',13X,'YN',16X,'TSOL',16X,'ERROR')
      END

C PRO
C THIRD- ORDER  RUNGE-KUTTA METHOD
C OUR PROBLEM IS :Y'=-Y,Y(O)=1
C THEORETICAL SOLUTION:Y(XN)=1/EXP(XN)
       DOUBLE PRECISION XN,YN,H,ONE,THREE,TWO,FOUR
      DOUBLE OPEN(6,FILE='RUNG2.OUT')
      PRECISION TSOL,ERROR,K1,K2,K3
      H=0.1D0
      YN=1.0D0
      XN=0.1D0
      HALF=0.5D0
      THREE=3.0D0
```

```
      ONE=1.0D0
      TWO=2.0D0
      FOUR=4.0D0
      WRITE(6,101)
3     K1=-YN
      K2=-(YN+(H/THREE)*K1)
      K3=-(YN+(H/THREE)*TWO*K2)
      YN=YN+H/FOUR*(K1+THREE*K3)
      TSOL=ONE/EXP(XN)
      ERROR=TSOL-YN
      WRITE(6,100)XN,YN,TSOL,ERROR
      XN=XN+H
      IF(XN.LE.ONE) GOTO 3
 100  FORMAT(D6.1,1X,3(3X,D19.13))
 101  FORMAT(2X,'XN',13X,'YN',16X,'TSOL',16X,'ERROR')
      END

C PRO
C SECOND-ORDER  RUNGE-KUTTA METHOD
C OUR PROBLEM IS :Y'=Y,Y(O)=1
C THEORETICAL SOLUTION:Y(XN)=EXP(XN)
      DOUBLE PRECISION XN,YN,H,ONE,TWO
      DOUBLE PRECISION TSOL,ERROR,K1,K2
      OPEN(6,FILE='RUNG2.OUT')
      H=0.1D0
      YN=1.0D0
      XN=0.1D0
      ONE=1.0D0
      TWO=2.0D0
      WRITE(6,101)
3     K1=YN
      K2=(YN+H*K1)
      YN=YN+H/TWO*(K1+K2)
      TSOL=EXP(XN)
      ERROR=TSOL-YN
      WRITE(6,100)XN,YN,TSOL,ERROR
       XN=XN+H
      IF(XN.LE.ONE) GOTO 3
100   FORMAT(D6.1,1X,3(3X,D19.13))
101   FORMAT(2X,'XN',13X,'YN',16X,'TSOL',16X,'ERROR')
      END
C PRO
C THIRD- ORDER  RUNGE-KUTTA METHOD
C OUR PROBLEM IS :Y'=Y,Y(O)=1
C THEORETICAL SOLUTION:Y(XN)=EXP(XN)
      DOUBLE PRECISION XN,YN,H,ONE,THREE,TWO,FOUR
      DOUBLE PRECISION TSOL,ERROR,K1,K2,K3
      OPEN(6,FILE='RUNG2.OUT')
      H=0.1D0
      YN=1.0D0
      XN=0.1D0
      HALF=0.5D0
      THREE=3.0D0
      ONE=1.0D0
      TWO=2.0D0
      FOUR=4.0D0
      WRITE(6,101)
3     K1=YN
      K2=(YN+(H/THREE)*K1)
      K3=(YN+(H/THREE)*TWO*K2)
      YN=YN+H/FOUR*(K1+THREE*K3)
      TSOL=EXP(XN)
      ERROR=TSOL-YN
      WRITE(6,100)XN,YN,TSOL,ERROR
      XN=XN+H
```

```
      IF(XN.LE.ONE) GOTO 3
100   FORMAT(D6.1,1X,3(3X,D19.13))
101   FORMAT(2X,'XN',13X,'YN',16X,'TSOL',16X,'ERROR')
      END

C PRO
C SECOND-ORDER  RUNGE-KUTTA METHOD
C OUR PROBLEM IS :Y'=Y**2,Y(O)=1
C THEORETICAL SOLUTION:Y(XN)=1/(1-X)
      DOUBLE PRECISION XN,YN,H,ONE,TWO
      DOUBLE PRECISION TSOL,ERROR,K1,K2
      OPEN(6,FILE='RUNG2.OUT')
      H=0.1D0
      YN=1.0D0
      XN=0.1D0
      ONE=1.0D0
      TWO=2.0D0
      WRITE(6,101)
3     K1=YN**2
      K2=(YN+H*K1)**2
      YN=YN+H/TWO*(K1+K2)
      TSOL=ONE/(ONE-XN)
      ERROR=TSOL-YN
      WRITE(6,100)XN,YN,TSOL,ERROR
      XN=XN+H
      IF(XN.LE.ONE) GOTO 3
100   FORMAT(D6.1,1X,3(3X,D19.13))
101   FORMAT(2X,'XN',13X,'YN',16X,'TSOL',16X,'ERROR')
      END

C PRO
C THIRD- ORDER  RUNGE-KUTTA METHOD
C OUR PROBLEM IS :Y'=Y**2,Y(O)=1
C THEORETICAL SOLUTION:Y(XN)=1/(1-X)
      DOUBLE PRECISION XN,YN,H,ONE,THREE,TWO,FOUR
      DOUBLE PRECISION TSOL,ERROR,K1,K2,K3
       OPEN(6,FILE='RUNG2.OUT')
      H=0.1D0
      YN=1.0D0
      XN=0.1D0
      HALF=0.5D0
      THREE=3.0D0
      ONE=1.0D0
      TWO=2.0D0
      FOUR=4.0D0
      WRITE(6,101)
3     K1=YN**2
      K2=(YN+(H/THREE)*K1)**2
      K3=(YN+(H/THREE)*TWO*K2)**2
      YN=YN+H/FOUR*(K1+THREE*K3)
      TSOL=ONE/(ONE-XN)
      ERROR=TSOL-YN
      WRITE(6,100)XN,YN,TSOL,ERROR
      XN=XN+H
      IF(XN.LE.ONE) GOTO 3
100   FORMAT(D6.1,1X,3(3X,D19.13))
101   FORMAT(2X,'XN',13X,'YN',16X,'TSOL',16X,'ERROR')
       END
```

## *References*


[1] Lambert, J.D (1973); “Computer Methods in ODE’s”, John Wiley Publications, New York.

[2] Jain, M. K (1971); “Numerical Solution of Differential Equations (2nd Edition)”, Wiley Eastern Limited, New Delhi.